Team 203-4: The Dumpster Devs

- Zoe Stewart
- David Dayan
- John Lee
- Makayla Johnson
- Quinn Stone
- Mitchell LaRocque

Application Name: Sports Stats Master - http://sportsstatsmaster.com

Application Description:

- The user will be able to interact with the website to fetch information about sports statistics, generate plots, save their own plots, and find other interesting sports things like running sports analytics algorithms that are processed via a web server. The application will accomplish this by using API calls with www.sports-reference.com website to then generate graphics of a certain player/team/league. The graphics can be implemented well because of the integration between the sports reference API and Python libraries like Pandas. The user may also generate other cool information like a "6 degrees of freedom" type relationship between different players by using graph theory. Scope will be refined, but will focus on basic functionality at first like a single sport, with certain visualization and analytics features. Non-priority features include a registration/login system for saving information and other sports analytics. The application will use Python and other supported libraries on the back-end to generate graphs and other information. The graphs may be interactive (TBD).

Vision Statement: Sports Stats Master is for (fantasy) sports fanatics who want to learn more about the granular player details in an intuitive format. Sports Stats Master is a database analytical tool where users can discern who the best players are via position/team/age, compare player stats, build a custom list of favorite players, and much more! Unlike ESPN Fantasy Sports, we have a unique algorithm to rank players and make it seamless to share player profiles/stats.

Version Control: https://github.com/CSCI-3308-CU-Boulder/203_4_F20

- Team Meeting Logs
- Milestone Submissions
- All project code/components

Development Method: Jira Link

- We will follow the Agile/scrum/kanban SDM, and we will manage our progress, sprints, and backlogs through Jira.

- We will determine our preliminary, intermediate, and final outcomes and construct a development process for each of them: this will include planning, development, testing, deployment, and maintenance. We will focus on functionality.
- We will follow a 2 week sprint timeline and will assign a “product owner” who will prioritize product development/fixes and generate a product backlog. We will communicate through our slack channels and maintain a kanban board to track user stories and tasks.
- We will also have frequent “stand-up” meetings to discuss scrum: what was done, what will be done, and what resources we will need from each other. A scrum master will be assigned for each sprint and everyone in the team will be involved in this job.
- After each sprint we will hold reviews and retrospectives to evaluate our pace, user story weighting, and discuss other structural/administrative tasks/issues. We will also hold review to test and deploy our finished work.

Communication Plan: Other than Zoom, where our team will meet face to face to discuss and deliberate issues, we will be primarily using Slack and GroupMe as our communication outlets. Slack will be used for more formal coding communication and resources while GroupMe will be reserved for quick, urgent messages. Within Slack we have different channels depending on the topic such as “project”, “random”, “resources”, and “weekly meeting.” We will also be using Google Drive to a capacity as a collaboration medium.

Meeting Plan: Outside of recitation, our team will meet with Surya on Wednesdays at 6:30 pm via Zoom and then again on Sunday at 6:30 pm for two hours. These are tentative cadences that may be adjusted throughout the semester if need be. We may also add other meetings to hold “stand-ups” as we ramp up our application’s development.

Proposed Architecture Plan:

- The application can be hosted on some web service (G suite, Heroku) and can use either Django or Flask as the back-end framework because they are simple and should play well with our selection of python as our programming language (the choice between Django and Flask will depend on how we want to store our data via noSQL or SQL--options include PostGresSQL and MongoDB, respectively). HTML/CSS will be used on the front end with the potential to use a front-end framework like Vue.js. We will create a web server with a database, though the specifics are not known yet. Much of the data processing will be done on the backend. Django, for example, will make implementing authentication for log-ins very easy

Use Case Diagram:

Actors:

- 1. Sports fans
- 2. Fantasy sports players
- 3. Sports betters
- 4. Developers

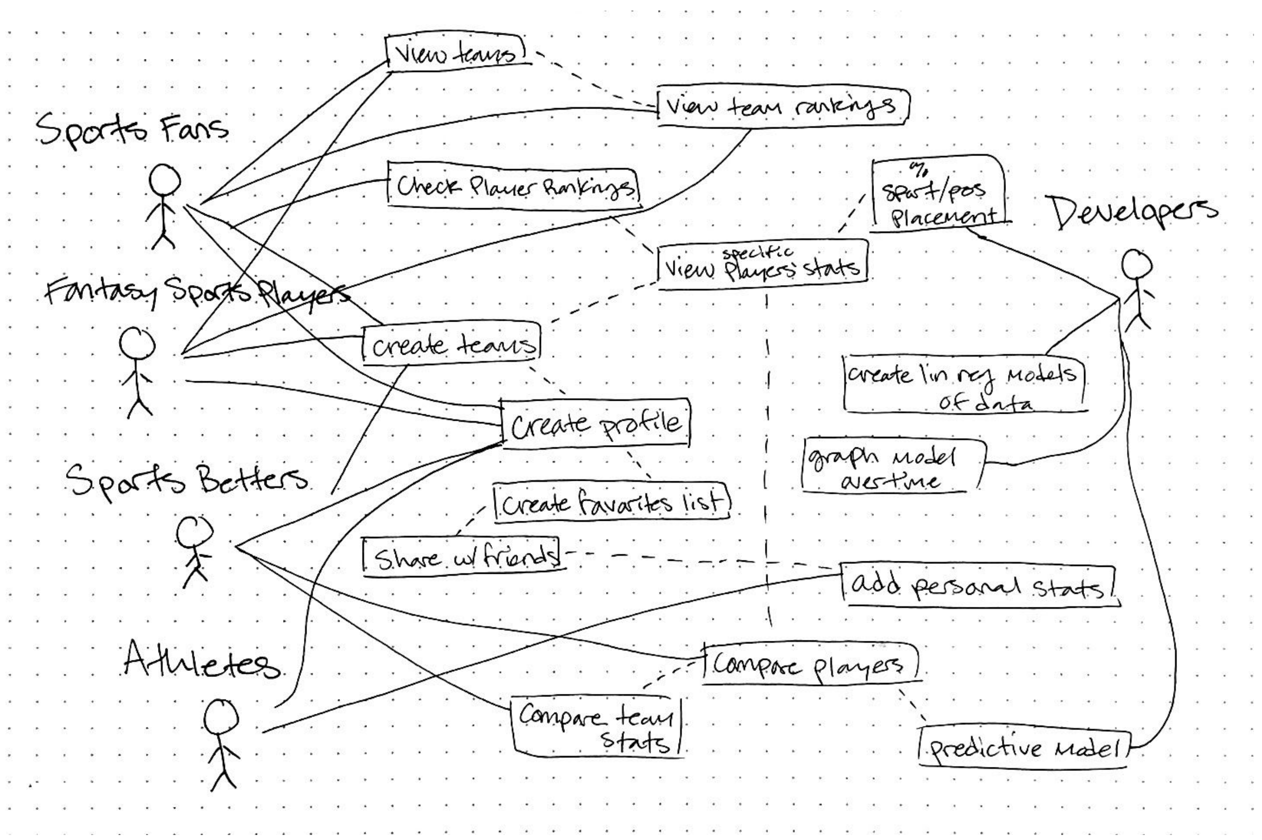
- 5. Athletes?

Use Cases:

- 1. All actors/Users create a profile used to login upon opening the site
- 2. Check a player’s rankings among others in that position
- 3. See granular player statistics with ease
- 4. Compare specific players using graphs/charts to see who is superior in each attribute
- 5. Create a custom list of favorite players that consistently updates
- 6. Users can share their favorite player stats with friends
- 7. Users can view entire teams, and sub rankings of that team
- 8. Athletes can enter and save their own stats to compare with players of their choice
- 9. Athletes can compare their personal stats with those of their friends?
- 10. Users (fantasy sports players?) can create their own teams to compare with friends
- 11. Compare players with a certain statistic (eg. all 6’ football players)
- 12. Create simple linear regression models (eg. height vs weight)
- 13. A predictive model for promising rookies and soon-to-be retirees
- 14. Compare different teams stats
- 15. See where a player places within their sport/position in terms of percentage (eg. top 2% 3-pointer)
- 16. Can create a graph modeling a player’s stats over time
- 17. Can compare how close other players are to another player based on criteria: “6 degrees of freedom” like receiving yards, points, championship wins.

Use Case Diagram:

Note: I definitely should've used a bigger piece of paper bc it got a little messy… sorry


View teams
View team rankings
Sports Fans
Check Player Rankings
% Sport/pos Placement
Developers
specific View Players' stats
Fantasy Sports Players
create teams
create lin reg models of data
Create profile
graph model over time
Sports Betters
Create favorites list
Share w/ friends
add personal stats
Athletes
Compare players
Compare team stats
predictive Model